



Agenda

1. Why CDISC Open Rules Matter
2. The Road to v1.0 – Progress So Far
3. What's Changed (and Why)
4. What's Next on the Roadmap
5. How to Get Involved



Why CDISC Open Rules Matter

... and why they matter to you ...

CDISC Conformance Rules and Regulatory Requirements

TODAY

Various source formats and structures

Specifications that are not machine-executable

Variety in interpretations of rule purpose

Individual implementations to executable code



Lack of consistency in data for regulatory submissions

GOAL

Aligned format and structure

Open-source machine-executable rules

One source of the truth, transparent and governed

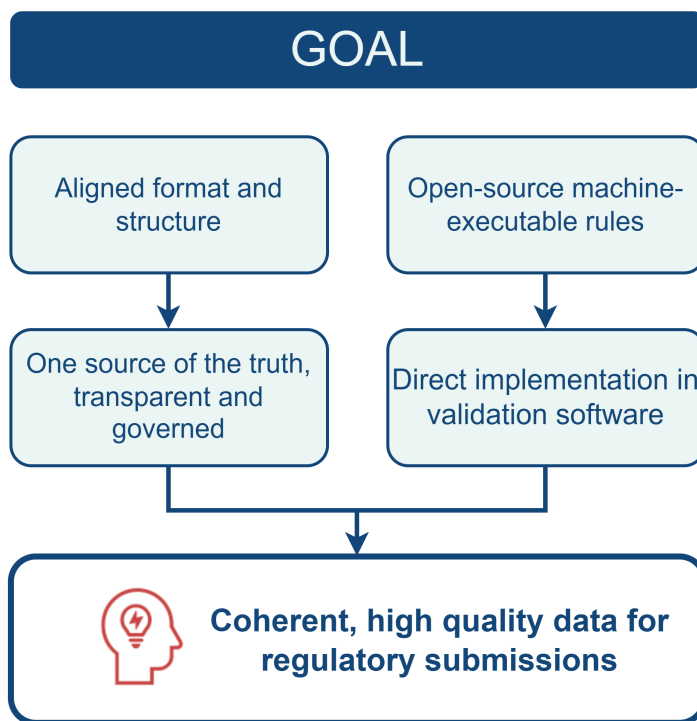
Direct implementation in validation software



Coherent, high quality data for regulatory submissions



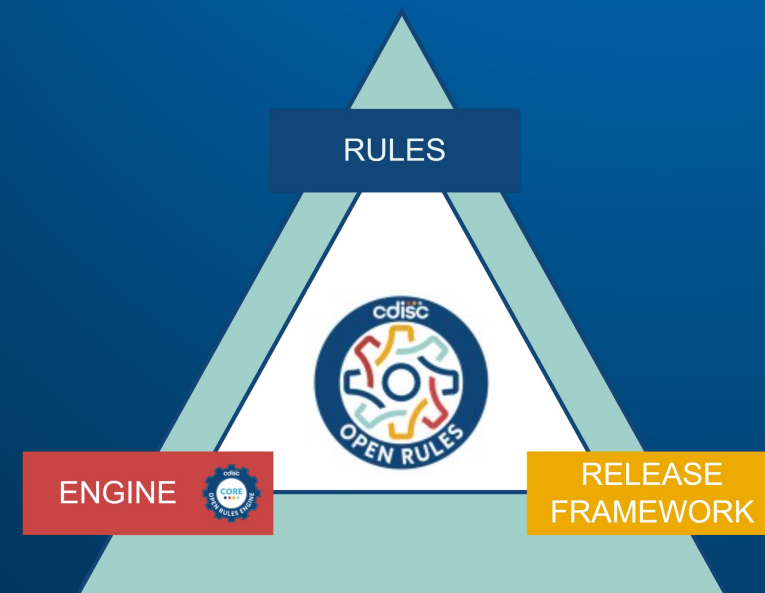
CDISC Conformance Rules and Regulatory Requirements





Components

- **Rules:** Authored in a simple, low-code format supported by a clear reference structure, enabling easy, transparent, and consistent creation and testing.
- **Engine:** CDISC Open Rules Engine (CORE) processes datasets and executes rules, providing a consistent open-source reference implementation.
- **Release Framework:** Ensures structured review, version control, and coordinated releases to maintain trust and consistency.



Available in CDISC Open-Source Alliance (COSA), released under MIT license



Additional Value for the Industry

- **Reusable Across Lifecycle:** Same conformance rules apply during setup, transfers, analysis, and packaging with automation and reduced manual work.
- **Interoperability:** Vendor-neutral rules provide a shared baseline logic that can be used consistently across tools and organizations.
- **Innovation:** Allows integration of company-specific validation checks and system extensions while remaining aligned with an industry-standard baseline.





The Road to v1.0 – Progress So Far



Rule Authoring Status for v1.0

Standard	# Rules	# Complete	% Complete
SDTMIG v3.2	411	376	91%
SDTMIG v3.3 and 3.4	505	461	91%
FDA Business Rules (SDTMIG)	186	103	55%
SENDIG v3.1.1*	312	222	71%
FDA Business Rules (SENDIG)	181	118	65%
Unique Rules	1139	884	78%
TIG v1.0**	484	409	85%
USDM v3.0 and v4.0**	260	257	99%
Unique Rules (Total)	1883	1550	82%

- *SDTMIG v3.2 and SENDIG v3.1.1 rules, not required to be complete for v1.0*
- *TIG v1.0 (partial rule set and only SDTM and SEND related rules) and USDM v3.0 and 4.0 are included in the scope of v1.0, but are separately funded*



Rule Authoring Status for v1.0

Standard	# Rules	# Complete	% Complete
SDTMIG v3.2*	411	376	91%
SDTMIG v3.3 and 3.4	505	461	91%
FDA Business Rules (SDTMIG)	186	103	55%
SENDIG v3.1.1*	312	222	71%
FDA Business Rules (SENDIG)	181	118	65%
Unique Rules	1139	884	78%
TIG v1.0**	484	409	85%
USDM v3.0 and v4.0**	260	257	99%
Unique Rules (Total)	1883	1550	82%

- Pending rules are mostly related to external dictionaries*
- Started from the FDA Business Rules = very high-level definition of a rule*



Engine – Current State (v0.15)

cdisc-rules-engine Public 9 Watch 30 Fork 90 Star

main 28 Branches 70 Tags

gerrycampion remove define_version (#1683) 1809190 · 5 days ago 1,739 Commits

File	Commit	Time
.github	add pull-request perm to check schema markdown (#1668)	3 weeks ago
TestRule	#1616: User-friendly errors for missing/wrong Datasets tab a...	last month
cdisc_rules_engine	remove define_version (#1683)	5 days ago
resources	#1654 added regex to is_inconsistent_across_dataset operati...	2 weeks ago
scripts	983 handle missing external dictionaries (#1611)	3 weeks ago
tests	remove define_version (#1683)	5 days ago
.flake8	fixes (#1619)	last month
.funcignore	Iss 315 move cdisc library conformance rules generator to Git...	3 years ago
.gitignore	1604 automated ci workflows should run on push to main (#...	last month
.pre-commit-config.yaml	Update .pre-commit-config.yaml (#1647)	last month
Dockerfile.build	#710 Add CORE logo icon to executable builds and update R...	3 months ago
LICENSE	Initial commit	4 years ago
MANIFEST.in	update all commands and docs (#1460)	4 months ago
PYPL.md	Cg0371 (#1201)	10 months ago

About
Open source offering of the cdisc rules engine

core

Readme
MIT license
Activity
Custom properties
90 stars
9 watching
30 forks
Report repository

Releases 58
v0.15 Latest on Feb 25
+ 57 releases

Deployments 500+
DEV 3 hours ago
PROD 5 days ago
+ more deployments

- Development included already 1739 commits
- 58 releases so far
- Solved 1726 tickets
- Custom rule schema available
- JSONata implemented
- ...

Key Opportunities

Engine Refactoring

Performance
Scalability
Sustainability

Improved Rule Authoring

Centralized, trackable workflow
Built-in automated Testing
Accessible for all volunteers

Foster Innovation

Lower Engineering Dependency
Decreased Maintenance
Author Flexibility
AI Assisted Rule Authoring

Time for Some Reflections...

Lessons learned from building an open, community-driven standard at scale

We tried our best with what we knew then. Now we know more.



Doing it right the first time is great. Doing it better the second time is progress.

If no one complained, either it was perfect... or no one was listening.



What's Changed (and Why)



Verisian's Engine Refactoring



Verisian's Engine Refactoring

- Complete internal refactor of the engine architecture to utilise SQL



Verisian's Engine Refactoring

- Complete internal refactor of the engine architecture to utilise SQL
- Minimal changes to existing CDISC rule logic, operations, or operators



Verisian's Engine Refactoring

- Complete internal refactor of the engine architecture to utilise SQL
- Minimal changes to existing CDISC rule logic, operations, or operators
- No additional setup or installation required*



Verisian's Engine Refactoring

- Complete internal refactor of the engine architecture to utilise SQL
- Minimal changes to existing CDISC rule logic, operations, or operators
- No additional setup or installation required*

**With pgserver the engine runs locally without any performance drop*

Engine Refactoring – Performance Improvements

Old Engine	Refactored Engine

Engine Refactoring – Performance Improvements

Old Engine	Refactored Engine
Failed to run datasets of 1 GB	

Engine Refactoring – Performance Improvements

Old Engine	Refactored Engine
Failed to run datasets of 1 GB	Successfully ran datasets > 15GB

Engine Refactoring – Performance Improvements

Old Engine	Refactored Engine
Failed to run datasets of 1 GB	Successfully ran datasets > 15GB
Memory-based implementation	

Engine Refactoring – Performance Improvements

Old Engine	Refactored Engine
Failed to run datasets of 1 GB	Successfully ran datasets > 15GB
Memory-based implementation	SQL-based implementation with near unlimited scaling

Engine Refactoring – Performance Improvements

Old Engine	Refactored Engine
Failed to run datasets of 1 GB	Successfully ran datasets > 15GB
Memory-based implementation	SQL-based implementation with near unlimited scaling
No alternative options	

Engine Refactoring – Performance Improvements

Old Engine	Refactored Engine
Failed to run datasets of 1 GB	Successfully ran datasets > 15GB
Memory-based implementation	SQL-based implementation with near unlimited scaling
No alternative options	Embedded alternative with equal capabilities; no additional software installation required

Engine Refactoring – Performance Improvements

Old Engine	Refactored Engine
Failed to run datasets of 1 GB	Successfully ran datasets > 15GB
Memory-based implementation	SQL-based implementation with near unlimited scaling
No alternative options	Embedded alternative with equal capabilities; no additional software installation required

The refactored engine removes dataset size and scalability limitations while improving accuracy and error reporting.

Rule Authoring

- Full review of the rules was done

Rule Authoring

- Full review of the rules was done
- Decision was made to
 - Move away from the old rule authoring method
 - Transfer to GitHub as source of truth for all CDISC Open Rules

Rule Authoring

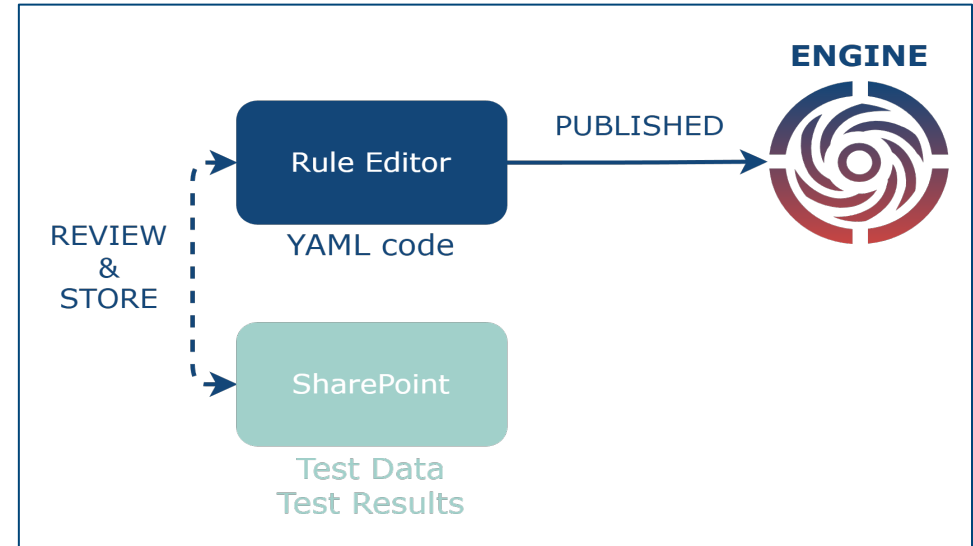
- Full review of the rules was done
- Decision was made to
 - Move away from the old rule authoring method
 - Transfer to GitHub as source of truth for all CDISC Open Rules
- GitHub keeps track of rule completion status and includes clear version control and review workflows

Rule Authoring

- Full review of the rules was done
- Decision was made to
 - Move away from the old rule authoring method
 - Transfer to GitHub as source of truth for all CDISC Open Rules
- GitHub keeps track of rule completion status and includes clear version control and review workflows
- Built a regression testing model to ensure changes do not affect previously validated rules

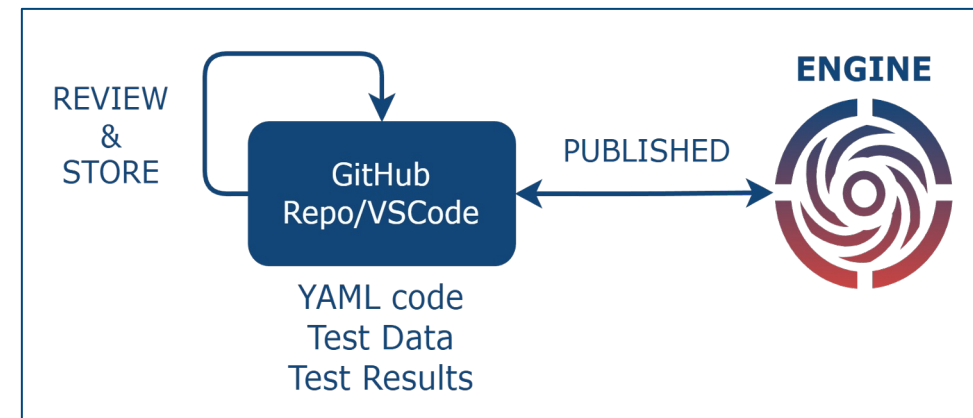
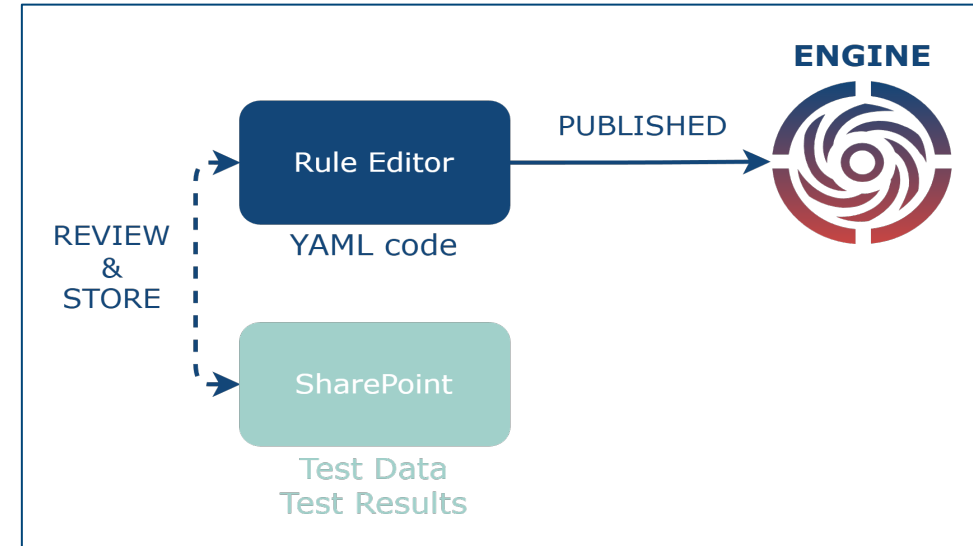
Rule Authoring

- Full review of the rules was done
- Decision was made to
 - Move away from the old rule authoring method
 - Transfer to GitHub as source of truth for all CDISC Open Rules
- GitHub keeps track of rule completion status and includes clear version control and review workflows
- Built a regression testing model to ensure changes do not affect previously validated rules



Rule Authoring

- Full review of the rules was done
- Decision was made to
 - Move away from the old rule authoring method
 - Transfer to GitHub as source of truth for all CDISC Open Rules
- GitHub keeps track of rule completion status and includes clear version control and review workflows
- Built a regression testing model to ensure changes do not affect previously validated rules



GitHub Rule Repository – What?

- Stores CDISC Open Rules YAML code, test data and validation results

core-contributor / rules / CORE-000001 /	
eljanssens CORE-000001 - corrected YAML code, updated test data and validation r... ✓	
Name	Last commit message
..	
negative/01	CORE-000001 - corrected YAML code, updated test data and
positive/01	CORE-000001 - corrected YAML code, updated test data and
CG0176,TIG0405.bc418819-cbaa-4513-89c9-9aeb3c7ce954.yml	CORE-000001 - corrected YAML code, updated test data and

GitHub Rule Repository – What?

- Stores CDISC Open Rules YAML code, test data and validation results

core-contributor / rules / CORE-000001 /	
eljanssens CORE-000001 - corrected YAML code, updated test data and validation r... ✓	
Name	Last commit message
..	
negative/01	CORE-000001 - corrected YAML code, updated test data and
positive/01	CORE-000001 - corrected YAML code, updated test data and
CG0176,TIG0405.bc418819-cbaa-4513-89c9-9aeb3c7ce954.yml	CORE-000001 - corrected YAML code, updated test data and

Traceable, transparent
and controlled

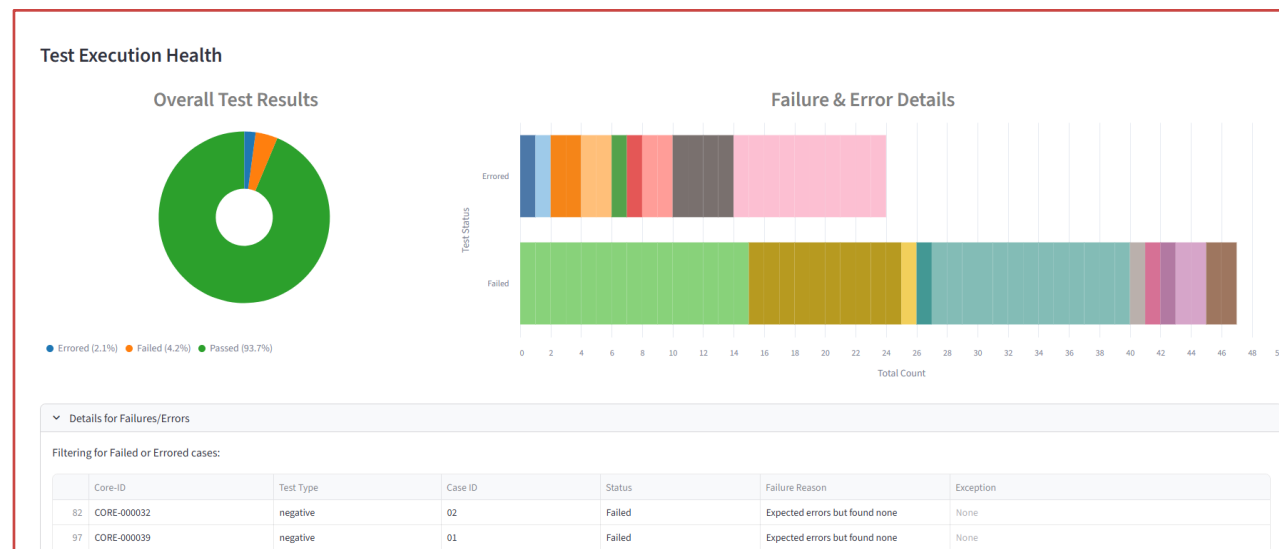
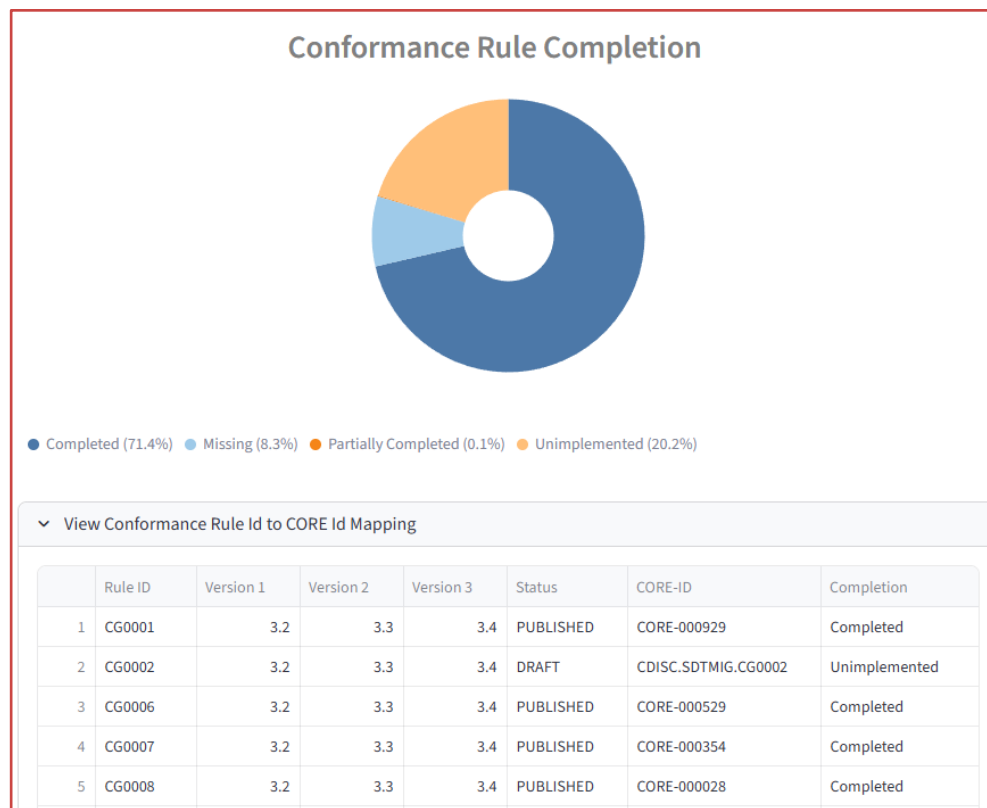
Writing and automated
testing is done in
VSCode

The screenshot shows the VS Code interface with a file explorer on the left and a code editor on the right. The file explorer shows a directory structure for 'CORE-000001' with subdirectories 'negative/01' and 'positive/01'. The code editor displays a YAML file named 'CG0176,TIG0405.bc418819-cbaa-4513-89c9-9aeb3c7ce954.yml'. The file content includes a 'Check' section with a rule for 'IECAT' and 'IEORRES', a 'Core' section with metadata, and an 'Outcome' section with a message and output variables. The 'Outcome' section shows a message: 'IECAT equals 'INCLUSION', but IEORES is not equal to 'N''. The 'Output Variables' section lists 'IETESTCD' and 'IEORRES'.

```
5 Authorities:
6   - Organization: CDISC
7   Standards:
53     - Name: TIG
54     Version: '1.0'
68 Check:
69   all:
70     - name: IECAT
71       operator: equal_to
72       value: 'INCLUSION'
73       value_is_literal: true
74     - name: IEORES
75       operator: not_equal_to
76       value: 'N'
77       value_is_literal: true
78 Core:
79   Id: CORE-000001
80   Status: Published
81   Version: '1'
82 Description: Raise an error when IECAT equals 'INCLUSION' but IEORES does
83   not equal 'N'.
84 Executability: Fully Executable
85 Outcome:
86   Message: IECAT equals 'INCLUSION', but IEORES is not equal to 'N'.
87   Output Variables:
88     - IETESTCD
89     - IECAT
90     - IEORES
91 Rule Type: Record Data
92 Scope:
93   Classes:
94     Include:
95       - FINDINGS
96   Domains:
97     Include:
98       - IE
99 Use Case: INDH
100 Sensitivity: Record
101
102
```

GitHub Rule Repository – What?

- Streamlit dashboard to track rule completion

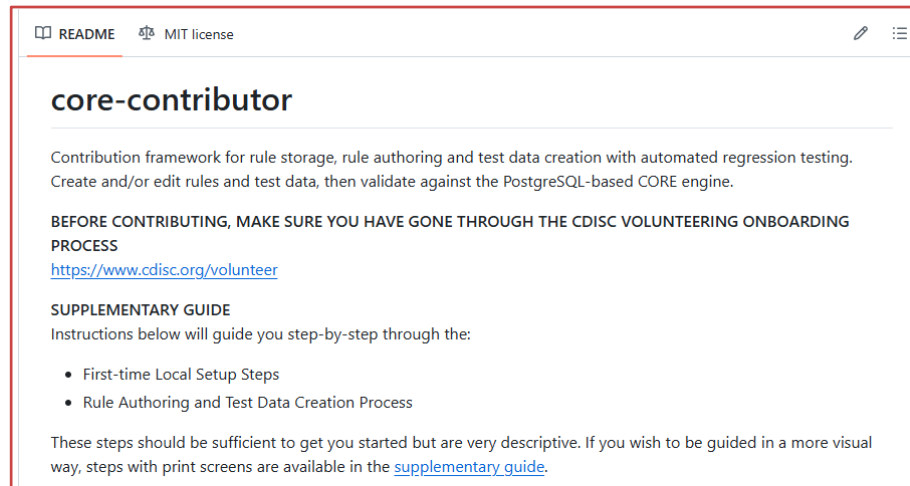


GitHub Rule Repository – Contributions Made Easy

- User-friendly setup scripts and a detailed guides
- Training in rule authoring

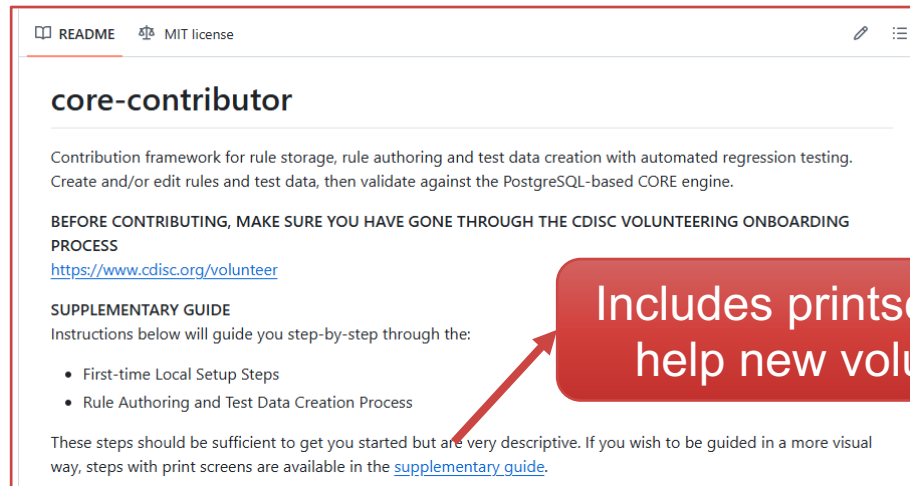
GitHub Rule Repository – Contributions Made Easy

- User-friendly setup scripts and a detailed guides
- Training in rule authoring



GitHub Rule Repository – Contributions Made Easy

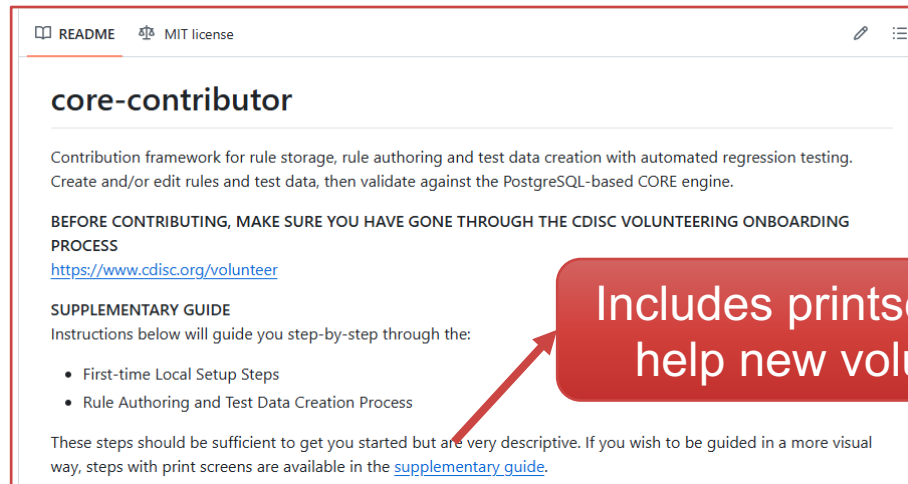
- User-friendly setup scripts and a detailed guides
- Training in rule authoring



Includes printscreens to help new volunteers

GitHub Rule Repository – Contributions Made Easy

- User-friendly setup scripts and a detailed guides
- Training in rule authoring



The screenshot shows the GitHub README for the 'core-contributor' repository. It includes a 'README' tab and a 'MIT license' link. The title is 'core-contributor'. The description states: 'Contribution framework for rule storage, rule authoring and test data creation with automated regression testing. Create and/or edit rules and test data, then validate against the PostgreSQL-based CORE engine.' Below this, it says 'BEFORE CONTRIBUTING, MAKE SURE YOU HAVE GONE THROUGH THE CDISC VOLUNTEERING ONBOARDING PROCESS' with a link to <https://www.cdisc.org/volunteer>. A section titled 'SUPPLEMENTARY GUIDE' says 'Instructions below will guide you step-by-step through the:' followed by a bulleted list: 'First-time Local Setup Steps' and 'Rule Authoring and Test Data Creation Process'. At the bottom, it says 'These steps should be sufficient to get you started but are very descriptive. If you wish to be guided in a more visual way, steps with print screens are available in the [supplementary guide](#).' A red arrow points from a red callout box to the 'supplementary guide' link.

core-contributor

Contribution framework for rule storage, rule authoring and test data creation with automated regression testing. Create and/or edit rules and test data, then validate against the PostgreSQL-based CORE engine.

BEFORE CONTRIBUTING, MAKE SURE YOU HAVE GONE THROUGH THE CDISC VOLUNTEERING ONBOARDING PROCESS

<https://www.cdisc.org/volunteer>

SUPPLEMENTARY GUIDE

Instructions below will guide you step-by-step through the:

- First-time Local Setup Steps
- Rule Authoring and Test Data Creation Process

These steps should be sufficient to get you started but are very descriptive. If you wish to be guided in a more visual way, steps with print screens are available in the [supplementary guide](#).

Includes printscreens to help new volunteers

First-time Local Setup Steps

IMPORTANT NOTE

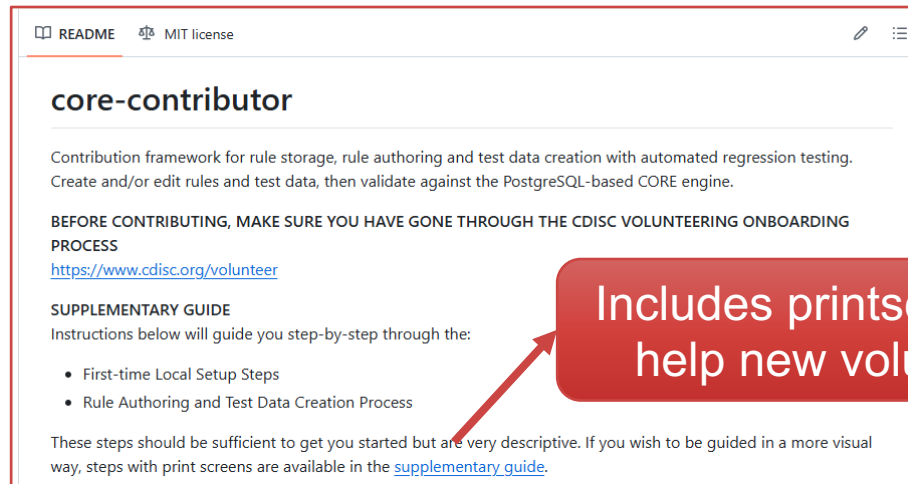
You may need your IT support team to install some of the following software for you. In particular, the setup script requires python3.12 to run properly. If you don't have it installed, the script will attempt to install it for you, but this is likely to be blocked by your company settings. If so, you will need to contact IT.

Follow steps 1 - 11 carefully.

1. Create a free GitHub account: <https://github.com/signup>
2. Install Git, following the instructions here: <https://git-scm.com/install>
 - When prompted, ensure you check the "Add to PATH" option (or select "Git from the command line and also from 3rd-party software")
 - Keep all other default settings throughout the installer
 - You DO NOT need to actually run Git as a program, so close any pop-ups that appear after the installation

GitHub Rule Repository – Contributions Made Easy

- User-friendly setup scripts and a detailed guides
- Training in rule authoring



Includes printscreens to help new volunteers

First-time Local Setup Steps

IMPORTANT NOTE

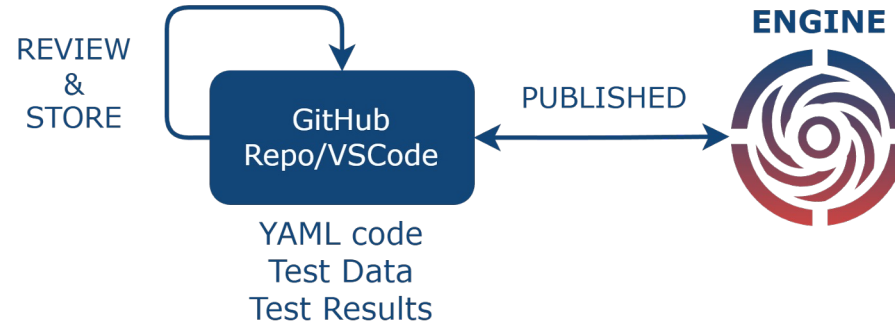
You may need your IT support team to install some of the following software for you. In particular, the setup script requires python3.12 to run properly. If you don't have it installed, the script will attempt to install it for you, but this is likely to be blocked by your company settings. If so, you will need to contact IT.

Follow steps 1 - 11 carefully.

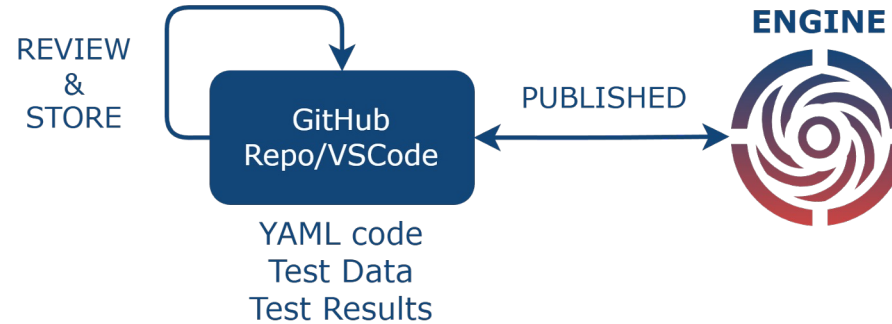
1. Create a free GitHub account: <https://github.com/signup>
2. Install Git, following the instructions here: <https://git-scm.com/install>
 - When prompted, ensure you check the "Add to PATH" option (or select "Git from the command line and also from 3rd-party software")
 - Keep all other default settings throughout the installer
 - You DO NOT need to actually run Git as a program, so close any pop-ups that appear after the installation

Easily write Rules and perform Unit Testing within the repository

Improved Rule Authoring Process



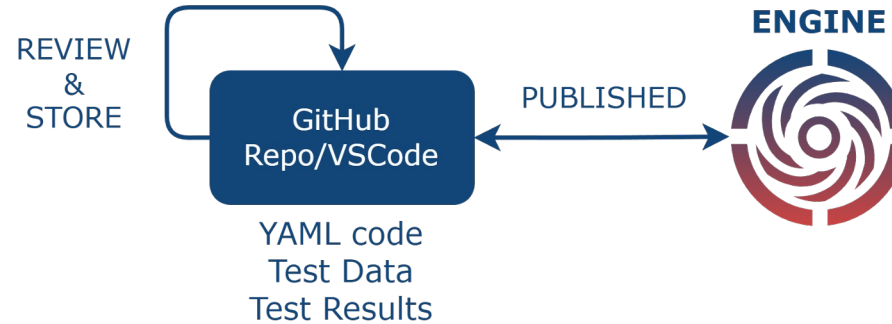
Improved Rule Authoring Process



Centralized and Trackable Workflow

- Git-based rule repository for storing, versioning, and tracking
- Streamlit dashboard for quick analysis and progress visibility

Improved Rule Authoring Process



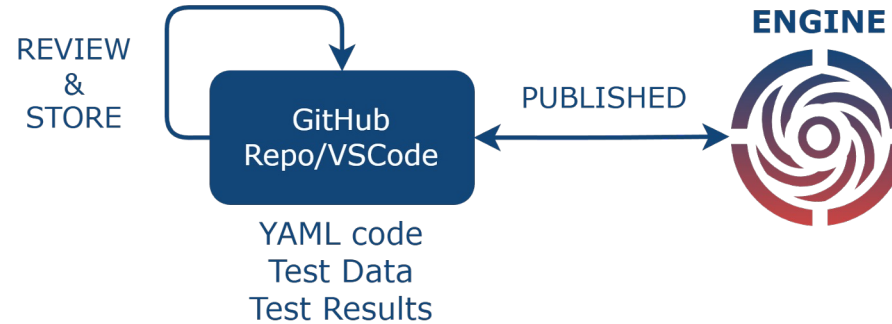
Centralized and Trackable Workflow

- Git-based rule repository for storing, versioning, and tracking
- Streamlit dashboard for quick analysis and progress visibility

Easy and Guided Contribution

- Simple setup scripts and clear contributor guide
- Accessible to all skill levels
- Structured review process

Improved Rule Authoring Process



Centralized and Trackable Workflow

- Git-based rule repository for storing, versioning, and tracking
- Streamlit dashboard for quick analysis and progress visibility

Easy and Guided Contribution

- Simple setup scripts and clear contributor guide
- Accessible to all skill levels
- Structured review process

Integrated Training and Testing

- Built-in training materials
- Automated testing within the repository

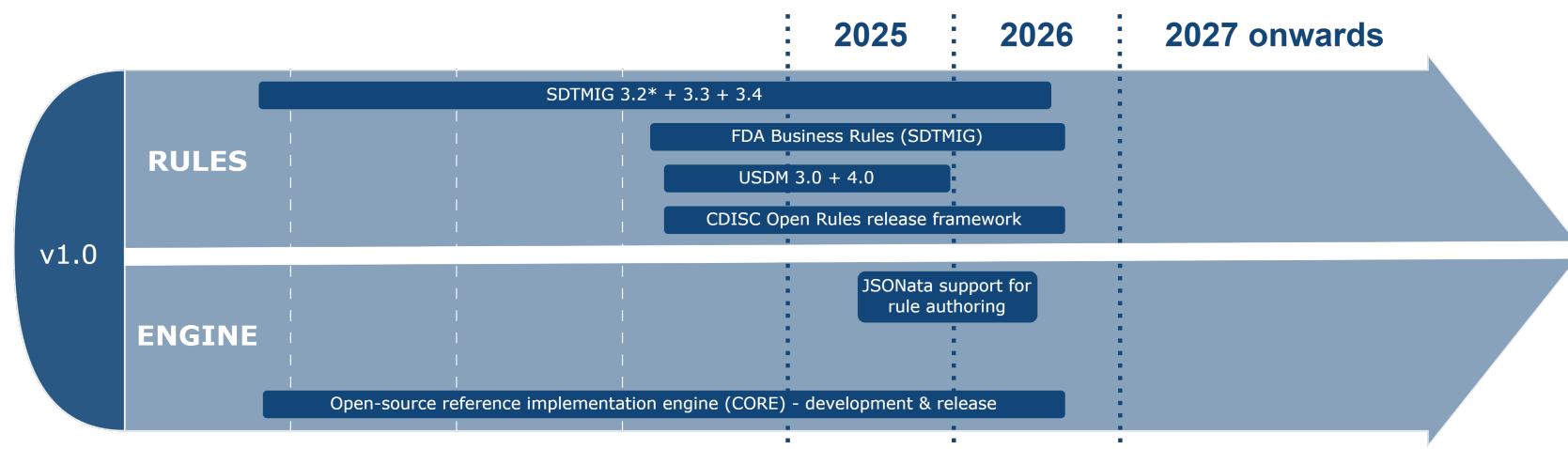


What's Next on the Roadmap

Roadmap

Note that Roadmap timelines are estimates, depend on volunteer-driven progress.

* SDTMIG 3.2 will be created but are not required to be complete for v1.0



Transferring CDISC Open Rules for SDTMIG and FDA Business Rules (SDTMIG) to GitHub Repository

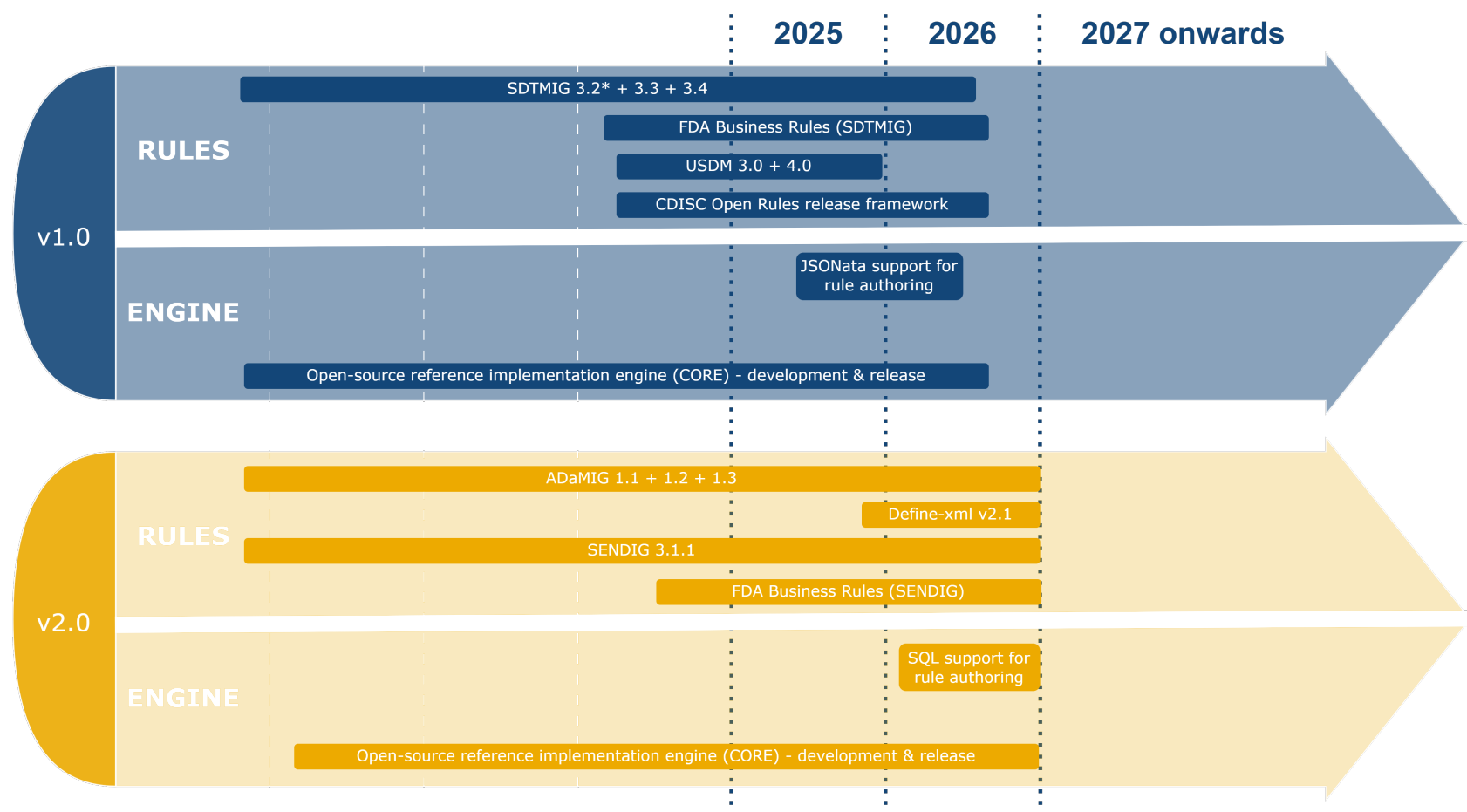
Further building the GitHub Repository for CDISC Open Rules for other standards

Roll out a release framework for the rules and the engine

Roadmap

***Note** that Roadmap timelines are estimates, depend on volunteer-driven progress.*

** SDTMIG 3.2 will be created but are not required to be complete for v1.0*



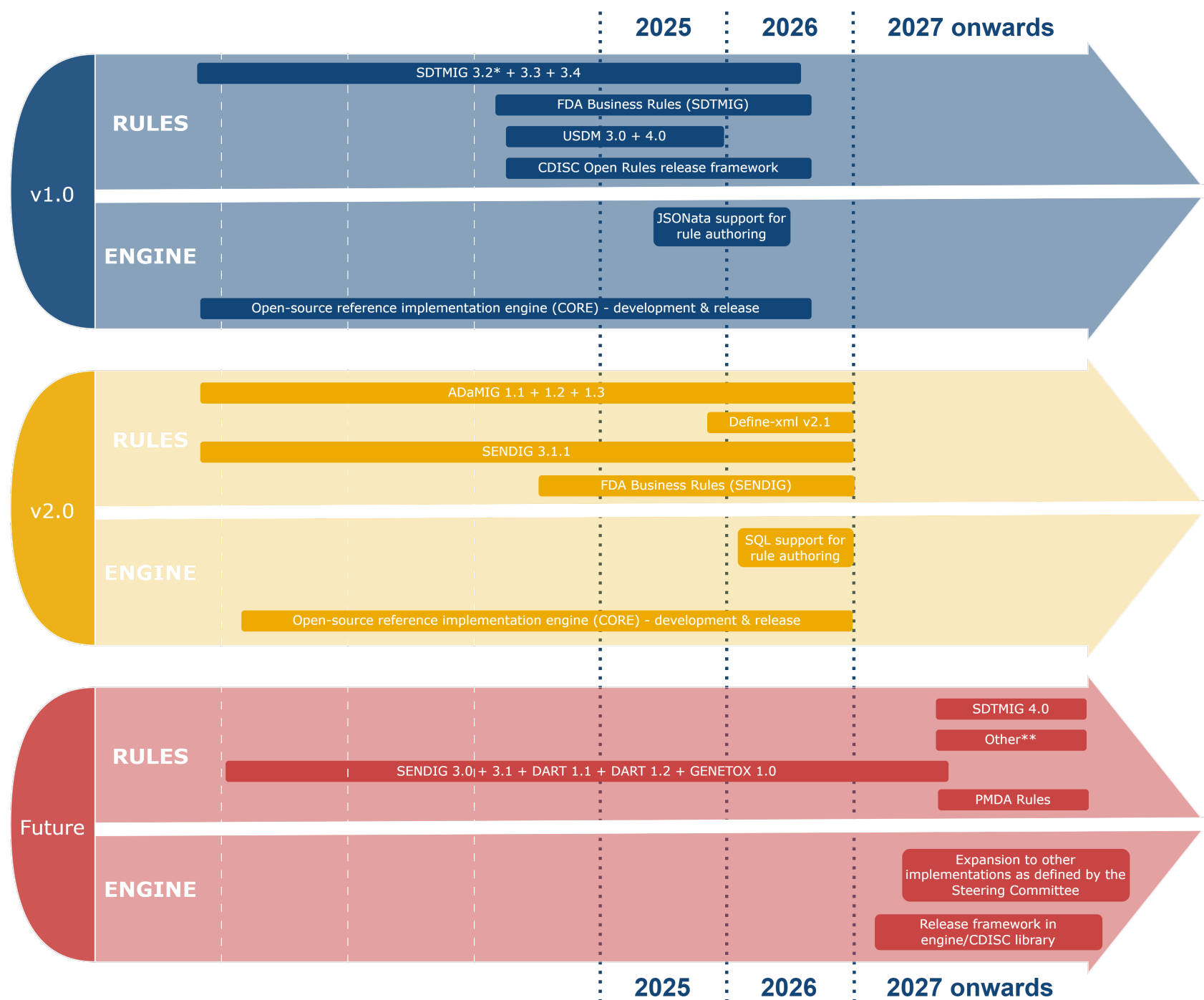
ADaM and define-xml experts are very much welcome for the next phase

Roadmap

***Note** that Roadmap timelines are estimates, depend on volunteer-driven progress.*

** SDTMIG 3.2 will be created but are not required to be complete for v1.0*

*** All remaining standards not regulatory required, updates to existing conformance rules, AI-driven rule authoring,...*





How to Get Involved



Volunteering

The success and completion of CDISC Open Rules is community driven

Rules need to be transferred to the GitHub Rule Repository, including a validation run



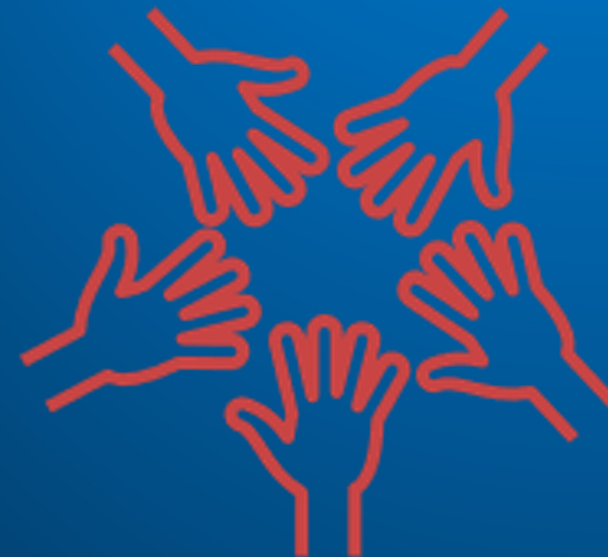
We currently have 13 contributors in our GitHub Rule Repository, successfully creating and validating CDISC Open Rules



Volunteering

The success and completion of CDISC Open Rules is community driven

**Join us in shaping the future of
CDISC Open Rules**



Sign up as a volunteer